

SYSTEM AND METHOD OF
NON-CENTRALIZED ZERO KNOWLEDGE AUTHENTICATION
FOR A COMPUTER NETWORK

RELATED APPLICATIONS

[0001] This application claims priority to U.S. application number 60/418,889, which is incorporated herein by reference.

BACKGROUND

[0002] Computer systems intercommunicate via computer networks. For example, a first computer system frequently communicates with a second computer system over a computer network to obtain information. The computer network may include many different communication media. In one example, the computer network is an Ethernet local area network ("LAN"). In another example, the computer network is a wireless LAN. Information stored on the first computer system is often sensitive such that access to the information must be restricted. Accordingly, the first computer system often requires that the second computer system be authenticated before allowing the second computer system to access the information. Access to the computer network may also be restricted, requiring any computer system wishing to join the computer network to be authenticated before communicating with other devices on the network.

[0003] Authentication typically utilizes an identification protocol that requires a computer system to identify itself with authority to access a restricted computer system. In one example, a first computer system may require a "password" from the second computer system to enable authentication. However, in situations where the communication between the first and second computer systems is monitored by a third computer system, the password may be obtained by the third computer system, allowing unauthorized access by the third computer system to the first computer system. Identification protocols that provide authentication without transmission of a secret password, known as a 'key', are therefore utilized. A zero-knowledge identification

protocol ("ZKIP") is one example of a protocol that provides authentication without transmitting the key, thereby preventing the key from being stolen and misused.

[0004] Typically, in a computer network that uses authentication, there is only one authenticator that stores keys used to authenticate requests from other computer systems. The use of a single authenticator, however, may result in access problems when the computer system running the authenticator fails, or where communications to the authenticator fail, for example. Where the authentication is for important data or services, failure of the authenticator may prevent access to the data or services. Further, the use of a single authenticator also causes congestion within the computer network as all authentication traffic is directed to a single location.

[0005] Where a computer network is highly scalable and dynamic it is important to authenticate each computer system as it attempts to access the computer network. A digital mobile telephone network is one example of a dynamic computer network. The digital mobile telephone network consists of multiple base stations that are networked together, each base station providing one or more cells for the digital telephone network. Each mobile telephone handset connects to, and disconnects from, these cells as the handset changes location. It is therefore important that any authentication process used within the cell network be as fast and efficient as possible. Typically, to meet speed requirements for a digital mobile telephone network, the authentication process is simplified, thereby making it less reliable and less secure, making the mobile telephone network highly susceptible to snooping by third parties.

SUMMARY OF INVENTION

[0006] United States Patent No. 4,748,668, titled Method, Apparatus and Article for Identification and Signature, is incorporated herein by reference.

[0007] In one aspect, a method provides non-centralized zero knowledge authentication within a dynamic computer network. The dynamic computer network includes two or more authentication agents that interact with prover agents within computers wishing to gain access to the computer network. Using a zero-knowledge authentication protocol, the prover is either authenticated, or not, without communication of a secret.

[0008] In another aspect, a software product (firmware, for example) is distributed with a hardware device to provide non-centralized zero-knowledge authentication. In one example, the hardware device is a router connected to a

network. The router communicates with a prover agent within a mobile computer (e.g., a laptop computer system or a mobile telephone handset) that seeks access to the network. Once the prover agent is authenticated and authorized, the router permits the mobile computer to access part of or the entire network.

[0009] In one aspect, methods are provided for authentication of identity or group membership. One such method involves zero-knowledge authentication. An authentication dialog between a verifying agent (“verifier”) and an agent to be verified (“prover”) is conducted without revealing information about a secret (“secret”) that is used to prove identity (or group membership without actually disclosing prover's identity). Authentication is achieved when verifier asks prover I -times ($I > 0$) to perform an action that can only be reliably performed by an entity that knows a secret. Prover answers verifier with results of action. If prover does not answer correctly, authentication is invalid. This challenge-response-validation iteration is repeated I -times to establish a sufficient level of probability that prover answered with knowledge of secret. One advantage of zero-knowledge authentication is inability for an eavesdropper to learn secret and steal means to prove identity to verifier. Another advantage is inability for verifier to later masquerade as a prover to a third-party.

[0010] In another aspect, methods are provided to allow for greater probability of correctly authenticating prover with fewer challenge-response-validation iterations. One such method allows prover to have a set, greater than two, of possible answers, as is provided by Fiat-Shamir protocol. For example, a prover that answers verifier correctly with a member of set $\{0, 1, 2, 3\}$ has a 25% chance of being incorrectly authenticated with one challenge-response-validation iteration. Following Fiat-Shamir protocol, prover will answer verifier with one of two possible answers $\{0, 1\}$ and thereby require two challenge-response-validation iterations to achieve the same level of authentication probability.

[0011] In another aspect, an authenticator agent require a prover agent to repeat an authentication protocol until a specified confidence level that a prover agent is correctly authenticated has been satisfied. For example, a confidence level of 99% may require 10 iterations, where a confidence level of 99.9999% may require 20 iterations.

[0012] In another aspect, methods are provided to validate agents without unique indicia. One such method allows agents to validate based on indicia that they

are within a category of agents who have knowledge of secret common to all authentic agents. An advantage of using non-unique indicia is elimination of overhead required to generate, maintain, and validate unique indicia

[0013] In another aspect, methods are provided to publish secret used to authenticate agents. One such method allows a trusted source to periodically update and publish the secret and product of two large prime numbers ("product"). The frequency of updates is less than the predicted length of time a malicious party could factor product or guess secret. Trusted source generates, encrypts, and publishes secret and product. Prover and verifier read encrypted values for secret and product, from trusted source, and use previous values of secret and product to decrypt new values for secret and product. Prover and verifier now have all information required to perform authentication processes.

[0014] One advantage of using methods described above is elimination of steps required to derive keys to encrypt and decrypt messages.

BRIEF DESCRIPTION OF DRAWINGS

[0015] FIG. 1 is a flowchart illustrating one process for generating and publishing secret and product of two large prime numbers;

[0016] FIG. 2 shows a method of decrypting secret and product of two large prime numbers;

[0017] FIG. 3 shows a challenge-response-validation iteration process between prover and verifier agent; and

[0018] FIG. 4 shows a system with three clients, each including a prover agent, and a host computer with a verifier agent.

[0019] FIG. 5 illustrates one system for providing non-centralized zero knowledge authentication within a dynamic computer network.

DETAILED DESCRIPTION OF DRAWINGS

[0020] FIG. 1 shows one method 10 for generating and publishing a secret and a product of two large prime numbers. Method 10 is, for example, implemented by a 'trusted' source as described below. In step 14, an initial value of secret s is generated from a seed value, and two large prime numbers (" p " and " q ") are randomly generated. Step 16 calculates a current product n' (n -prime) of the two large prime numbers p and q , and initializes previous product value n (n -not prime) as equal to n' .

In step 18, p and q are purged and made unreadable. In step 20, current secret number s' (s-prime) is generated to be a value relatively prime to n, greater than 0, and less than n. In step 22, values for encrypted secret s'' (s-double prime) and encrypted product of two large numbers n'' (n-double prime) are generated as: $s'' = s's \bmod n$, and $n'' = n's \bmod n$. In step 24, previous secret number s (s-not prime) is set equal to s' and n is set equal to n' . In step 26, values for n'' and s'' are published. At this point, publication process is complete and process 10 waits in step 28.

[0021] Values for s'' and n'' may become compromised by a malicious party that is able to factor or guess values. Therefore, the delay in step 28 terminates before values are likely to be compromised and process 10 is restarted at step 20 where a new s' is generated.

[0022] FIG. 2 shows one method 30 of decrypting secret s'' and product n'' . In step 34, an agent (e.g., a prover agent or an authentication agent) is created with an initial value for s and n. In step 36, the agent reads values of s'' and n'' published by the trusted source (e.g., method 10, FIG. 1). In step 38, values of s'' and n'' are decrypted by a modulus inverse operation. In step 40, the size of answer set (" t ") is used to determine a value (" v ") calculated as result of $s'^t \bmod n''$.

[0023] At this point, prover and verifier agents have data required to perform authentication. Because values for s'' and n'' published by trusted source periodically change, updated values for s'' and n'' will be retrieved. Step 42 is a delay based on a specific length of time or may be triggered at the start of an authentication process (e.g., a zero-knowledge identification protocol). After the delay in step 42, method 30 continues with step 36 and the agent will again contact the trusted source and read new values for s'' and n'' .

[0024] FIG. 3 shows a challenge-response-validation iteration dialog between a prover agent (shown as process 48) and a verifier agent (shown as process 50). Process 48 performs processing to establish a need to authenticate and begins zero-knowledge identification protocol 46 in step 52, which may include retrieving and decrypting current values of secret s'' and the product n'' . In step 52, process 48 (prover) sends a signal 54 to process 50 (verifier) to begin zero-knowledge identification protocol 46. In step 56, process 50 (verifier) performs any initial processing, which may include retrieving and decrypting current values of secret s'' and product n'' . In step 56, process 50 sends signal 58 to process 48 (prover) to begin the authorization process. In step 60, process 48 (prover) generates a random number

("r"). Random number r is then used, in step 62, to generate a number x such that $x = r^t \bmod n$. In step 62, process 48 sends a signal 64 containing x to process 50 (verifier). In step 66, process 50 (verifier) then calculates a reply value b as a member of set $\{0 \dots t-1\}$. In step 66, process 50 sends a signal 68 containing b to process 48 (prover). In step 70, process 48 (prover) uses b to calculate a number y such that $y = rs^b$. In step 70, process 48 sends a signal 72 containing y to process 50 (verifier). Step 74 in process 50 is a decision. In step 74, process 50 performs a test to determine if process 48 (prover) has passed this iteration of zero-knowledge identification protocol 46. If $y^t \bmod n = (xv^b) \bmod n$ and $y \neq 0$, then process 50 continues with step 78; otherwise process 50 continues with step 76. Step 78 in process 50 is a decision. In step 78, the number of challenge-response-verification iterations is compared to the number of iterations required to establish a suitable probability of correct authentication. If the number of challenge-response-validation iterations performed is the same as the number of challenge-response-validation iterations required, and process 48 (prover) has not failed any iterations, then process 48 continues with step 82; otherwise process 50 signals process 48 to continue with step 60, thus beginning another challenge-response-validation iteration by repeating steps 60 through 74.

[0025] In step 82, process 50 continues with processing appropriate for authenticated process 48 (prover) and process 50 terminates. In step 76, process 50 (verifier) continues processing as appropriate for non-authentic agents, and process 50 terminates.

[0026] FIG. 4 shows a system 89 with three clients 90(1-3), each running a prover agent 91(1-3), and a host 92 running an authentication agent 96 (verifier). Prover agents 91(1-3) implement process 48, FIG. 3, for example. Authentication agent 96 implements process 50, FIG. 3, for example. Communication links 100(1-3) establishes connectivity between clients 90(1-3) and a connection module 94 within host 92. In system 89, client 90(1) seeks access to secure area 98 of host 92. Communication link 100(1) establishes connectivity between client 90(1) and connection module 94 within host 92. In one example, communication link 100(1) is a telephone dial-up connection. In another example, communication link 100(2) is an Internet connection. In another example, authentication agent 96 (verifier) protects secure area 98 allowing access only to authenticated clients. Communication link 100(3) is an Ethernet LAN connection. After client 90(1) is authenticated by host

(92), a connection is established and client 90(1) is allowed access to secure area 98. Once this connection has been established, authentication agent 96 may distribute a new secret from trusted source 106 to prover agent 90(1) for use in future authentication dialog. When prover agent 90(1) requests authentication at a future time after connection 110 has been broken, authentication agent 96 requests credentials from prover agent 90(1) from trusted source 106 via the hosts internal connection 104. At this point the authentication dialog may take place between client 90(1) and host 92 to reestablish a trusted connection.

[0027] Zero-knowledge identification protocol 46, FIG. 3, is then performed. If zero-knowledge identification protocol 46 is successful, an access link 108 is activated to secure area 98, and client 90(1) may proceed with further processing. If zero-knowledge identification protocol 46 is not successful, processing continues with knowledge that client 90(1) is not authorized and, at a minimum, client 90(1) is inhibited from access to secure area 98.

[0028] FIG. 5 shows one system 500 that provides non-centralized zero-knowledge authentication within a dynamic network. Illustratively, system 500 includes two Ethernet LANs 502 and 504 that are not co-located. LAN 502 is connected to LAN 504 via a communication apparatus 505 that contains connection units 506, 508 and a communication link 510. Connection units 506 and 508 are, for example, routers or microwave transceivers. Communication link 510 is, for example, an ISDN link, the Internet, or a microwave link that provides data communication between two remote locations.

[0029] LAN 504 is shown connected to a wireless LAN device 512 that provides wireless connectivity to mobile computers 514 and 516. LAN 504 also illustratively connects to computer system 518 that includes authentication agent 520 (verifier). Before mobile computer 514 connects to LAN 504, it is first authenticated using zero-knowledge identification protocol 46 as shown in FIG. 3. Mobile computer 514 includes a prover agent 522 that interacts with authentication agent 520 to perform zero-knowledge identification protocol 46. Mobile computer 516 includes a prover agent 524 that interacts with authentication agent 520 to gain authentication to access LAN 504.

[0030] Trusted source 106, FIG. 4, implements process 10, FIG. 1, to generate a new secret s' and a new product n' periodically to prevent the malicious party compromising the values by guessing or factoring. Thus, once computer system 536

has been authenticated and is connected to LAN 502 it receives new values for secret s" and product n", using an encrypted message based on its current values for secret s" and product n". Thus, integrity and security of system 500 is maintained at a high level. Only during initialization of system 500, or when a mobile computer (e.g., mobile computers 514, 516) connects to wireless LAN interface 512 and requests authentication, is a predefined secret used.

[0031] Computer system 530 illustratively connects to LAN 502 and includes authentication agent 532 (prover). Computer systems 534 and 536 also connect to LAN 502; computer system 534 includes a prover agent 538 and computer system 536 includes a prover agent 540. Prover agent 538 interacts with authentication agent 532 to authenticate computer system 534 for access to LAN 502. Similarly, prover agent 540 interacts with authentication agent 532 to authenticate computer system 536 for access to LAN 502.

[0032] Authentication agents 520 and 532 operate independently to authenticate mobile computers 514, 516 and desktop computers 534, 536 for access to LANs 504 and 502, respectively. Optionally, once a computer (e.g., computers 534, 536 and mobile computers 514 and 516) is authenticated and remains connected within system 500, it may operate to authenticate other computers (i.e., may operate as an authentication agent). Further, once authenticated and connected within system 500, the computer may operate to interact with other computers seeking authentication, enabling communication between the other computers and an authentication agent.

[0033] For example, and with reference to FIG. 5, consider computer 518 and 530 existing on the computer network defined by LANs 502, 504 and communications link 510 (at boot up to establish the network, computer 518, 530 are initialized with the same secret and thus both operate with respective authentication agents, as shown). When any other computer 534, 536, 514, 516 desires access to this computer network, it may do so only through zero knowledge authentication, such as zero-knowledge identification protocol 46 (i.e., the dialog between authentication agent 520 and prover agent 538, 540, 522, 524, respectively). Once authenticated on the network, the computer may be promoted to operate with an additional authentication agent so as to provide authentication to other computers desiring access to the network. Accordingly, the network is "dynamic" in that it allows additional, flexible authentications to occur and expand the network. To enable this non-

centralized zero knowledge authentication, authentication software (including authentication and prover agents) may be preloaded into each computer (e.g., computers 514, 516, 518, 530, 534, 536).

[0034] In one example, a computer network includes multiple base stations that operate to provide a mobile telephone network. Each base station contains an authentication agent. Each mobile handset includes a prover agent that connects to the mobile telephone network. Before the mobile handset is allowed to use any services of the mobile telephone network, the authentication agent in the base station selected by the mobile handset interacts with the prover agent in the mobile handset. If the authentication agent is satisfied that the prover knows the secret, it becomes authenticated and authorized to use the mobile telephone network. By using a ZKIP, the secret is never transmitted to or from the mobile handset, and therefore not susceptible to malicious snooping.